(FILE 'HOME' ENTERED AT 15:08:12 ON 12 MAY 2000)

FILE 'INPADOC' ENTERED AT 15:08:14 ON 12 MAY 2000

FILE 'USPATFULL' ENTERED AT 15:08:19 ON 12 MAY 2000

L1 82255 S 395/?/NCL OR 364/?/NCL OR 708/?/NCL OR 709/?/NCL OR 710/?/NCL OR 370/?/NCL OR 707/?/NCL OR 345/?/NCL OR 701/?/NCL

L2 11 S (SEARCH? OR QUERY?)(P)(SELECT? OR CHOOS?)(P)(STATEMENT OR SENTENCE)(2P)(RESOURCE(A)LOCATOR OR URL)

L3 58 S (SEARCH? OR QUERY?)(P)(SELECT? OR CHOOS?)(P)(EXPRESSION)(2P)(RESOURCE(A)LOCATOR OR URL OR ADDRESS)

L4 25 S L3 AND L1

L5 130 S (SEARCH? OR QUERY?)(P)(SELECT? OR CHOOS?)(P)(STATEMENT OR SENTENCE OR EXPRESSION)(2P)(RESOURCE(A)LOCATOR OR URL OR ADD

L6 66 S L5 AND L1

L7 50 S 707/?/NCL AND L6

L26 ANSWER 2 OF 2 USPATFULL
PI US 5862325 19990119
SUMM This same problem of efficient information distribution is exacerbated in many-to-many communications relationships, such as among the members of a workgroup. Here, communications are much more frequent and timely, and there is much greater quantity of information to be shared, stored, archived, and indexed. Members of a workgroup also have a strong need to employ communications for group coordination, such as scheduling meetings, **conference** calls, project deadlines, etc. These communications involve time deadlines and feedback requirements which are not typically present in one-to-many communications relationships. 21/8-29

DETD As illustrated in FIG. 1, there are two principal methods for information transfer in a data communications system, both of which can operate through the Internet. First, a "**pushing**" **method** transfers information from the provider computer 1 directly to a known consumer computer 2. An example of such a system is **e-mail**. So long as the consumer's address is known, the information can be routed through the communications network directly to that recipient. For the **pushing method**, the provider must know the consumers who want to receive the information. The provider must also know how to address those recipients in the communications network. 12/47-58

DETD Appropriate programs executing on the provider computer 1 and the consumer computer 2 perform the functions necessary to transfer, maintain, and update the information at both locations. A program represents a set of stored instructions which are executed in a processor of the computer to process data, transmit and receive data, and produce displays. The provider program 12 operates to transmit changes in information stored in the provider database 11 at the provider computer 1. When changes are made to the information and the database, the provider program 12 operates to disseminate the changed information through the communications network 3. In the **pushing method**, the provider program 12 transmits the changed information, for example through **e-mail**, to the consumer computers 2 of all intended recipients. In the pulling method, the changed information is stored on a distribution server 32, such as a web server, which then can be accessed by the consumer computer 2. Any type of distribution server may be used, including network file servers, FTP servers, gopher servers, and so on. The type of distribution server used is not a limiting feature of the invention. The consumer program 22 will typically poll the distribution server 32 to determine whether the information has changed. This polling operation can be as simple as issuing a Web server HTTP file date request and comparing this with the file date of the last update. Polling is controlled by the information transferred from the provider program to the consumer program as further described below. Upon receipt of changed information, the consumer program 22 operates to perform certain functions with regard to that changed information. Principally, the information is stored in consumer database 21 on the consumer computer 2 for future reference and usage in controlling and automating communications between the consumer and provider. Furthermore, the information may be presented to a user at the consumer location, so that the user will be notified of the changed information. The information can be presented in a number of manners, including display or printing by the consumer program, sending an **e-mail** or voice-mail message to the user, paging the user, and other notification methods. 13/6-45

DETD In the most general case, the provider knows what communications

networks, network addresses, languages, encoding formats, data structures, and other communications processing data and methods are supported by the provider. Thus, the provider can include in the transferred information the data, metadata, and instructions necessary to control and coordinate general communications from the consumer to the provider or to parties related to the provider. For example, data, metadata and instructions in the transferred information can be used by the consumer program 22 or other computer programs running on the consumer computer 2 to automatically format, compress, encrypt, address, and transmit copies of a word processing document, spreadsheet, database or database query, or other computer file format. Corresponding data, metadata, and instructions in the provider program 12 can control and automate the reception of the received message, including decryption, decompression, notification of the provider, and acknowledgment of receipt to the consumer. The same control technique can be applied to the execution of real-time communications, such as telephone calls, **videoconferencing**, or whiteboard applications. 14/33-54

DETD The Recipient class 120 is used to determine the distribution of a communications object. Each communications object 110 is associated with one or more recipients 120 who receive an instance of the object when it is first created or when changes are made to it. Recipients are of two types: consumer programs 22, or distribution servers 32. A distribution server 32 may also be represented by a distribution service object. Distribution service objects are further discussed below. Transfer of communications objects 110 to both types of recipients is typically via the **push technique**. However recipients may also be tracked in the provider database 11 even if they use the pull technique of updating via the use of receipt acknowledgment messages. Acknowledgment messages are further described below. The **push method** may involve a fully automated transfer via a communications network 3, or it may involve a manual transfer such as a file copy over a network or via a computer floppy disk. Recipient objects 120 include the attributes necessary to generate and transmit an instance of the communications object to the recipient. To uniquely identify recipients even when names change, a SystemID attribute can used in addition to a Name attribute. System IDs are discussed below. Other attributes include the recipient's communications network address, such as an **e-mail** address, the type of encoding that should be used (e.g. MIME, BinHex, UUencoding, etc.), and the maximum attachment file size the recipient can accept (to determine if multiple attachments need to be sent). Recipients 120 have an association with methods 141 in order to allow different methods to be assigned to different recipients. An example is the communications object's update method. Communications objects transmitted to consumers via **e-mail** push may use one update method, while those transmitted to distribution servers may use a pull update method. Encoding methods, transmission methods, and other recipient-specific methods may also be assigned in this manner. 22/29-56

DETD FIG. 8 illustrates transfer of an object through **e-mail** using the **push technique**. The browser program 50 is not used for this function. The object may be attached as a MIME object to an **e-mail** message 38. Other attachment or encoding types may be used, such as BinHex or UUencoding. The object may also be encoded in ASCII within the text of the **e-mail** message itself. The optimal encoding method for each recipient can be selected and employed automatically by the provider program 12 when this information is included in the Recipients (120, FIG. 3) class, as further described below. The transmission steps for each attachment or encoding type may vary slightly. The transmission steps for a MIME attachment will be described here. The **e-mail** message is sent in the ordinary manner, using whichever **e-mail** servers and intermediaries are available (i.e., through the Internet 3), to reach the consumer's **e-mail** server 31. The consumer's **e-mail** program 62 retrieves the mail message from its server in the ordinary manner. Depending upon operation of the **e-mail** program, the attachment may be downloaded for storage in either an

internal or external MIME directory 63, 64, or left for storage on the **e-mail** server 31. The consumer program 22 then periodically polls the MIME directory 65, 66 or the **e-mail** server 31 to locate objects of a communications object MIME type. If a communications object type is located, it is read from the storage location and processed by the consumer program as described below. It may also be deleted from the MIME storage area by the consumer program 22 after it has been read and processed.

DETD After encoding, the communications object is transmitted to the recipient according to the attributes and methods of the recipient (steps 550-551). As discussed previously, according to a preferred embodiment, objects sent directly to consumer computers 2 using the **push method** are sent as **e-mail** messages or message attachments to the addresses of the recipients. These transmissions can be queued using scheduled events 117 to reduce system load. Objects sent to a distribution server 32 for distribution using a pull method are saved to the appropriate web server document directory. Alternatively, based upon the access the provider has to the provider's web server, the object could be mailed to the Web server administrator, uploaded as an HTTP form to the Web server, or otherwise stored for later posting by the Web server administrator. The transmission steps could also include an **e-mail** message, voicemail message, or other notification to the administrator that the object is ready to be stored on the server. Alternatively, transmission to a distribution server 32 can be automated through the use of a distribution service object. Distribution service objects will be further discussed below.

DETD Acknowledgment messages can still be used even when the distribution method uses the pull technique. This is accomplished identically to the above except for the following. First, the acknowledgment message object instance (810, FIG. 17) returned to the provider program 12 includes such additional data about the consumer as is necessary to create a recipient record 120. Second, if the recipient record 120 instance does not exist in the provider database 11, the SendAck method needs to create it, and also create the association 121 between the recipient 120, the communications object 110, and one or more methods 141, including an update method. This specialized use of an acknowledgment message object 110 is referred to as a registration message. Registration messages are important for three reasons. First, registration messages can be used to track communications object distribution and usage even when the provider does not have the capability to distribute updates using the **push technique**. An example is when an expensive, high-powered web server is used for high-volume distribution of a communications object, but an inexpensive personal computer and email account is used for tracking communications object acknowledgment messages. Second, registration messages can be used on an intermittent basis by only including the SendAck receipt method in selected communications object updates. This allows distribution statistics and other data to be gathered periodically rather than with every update. Third, if the acknowledgment message object 110 includes the **e-mail** address of the consumer, the resulting list of recipients 120 created by registration messages can allow the provider to convert the communications object update method from pull to push. Conversion between update methods is discussed further below.

DETD When a communications object instance is distributed using the **push technique**, updates are pushed by the provider program 12, so an update method is not required in the communications object. However, an update method may still be employed in this case for error correction. For example, if the provider typically distributes communications object updates via the **push technique** every 30 days, the provider could include in the communications object a receipt method that creates a scheduled event instance (117, FIG. 3) in the consumer program 22 to be activated in 60 days. With each subsequent update of the communications object, the receipt method would reschedule this scheduled event instance for another 60 days into the future. If a push transmission from the provider did not reach the consumer within a

60 day period, the scheduled event instance would be actived. It would trigger the update method which would send a message object 110 back to the provider program 12. This message object could contain the **e-mail** address of the consumer computer 2, the version and date of the last communications object received, and other such data as would allow the provider program 12 and consumer program 22 to resynchronize after an error condition.

DETD In certain cases it may be advantageous to combine both the push and the pull techniques of updating for a single communications object 110. For example, a provider may wish to use pull updating for distribution of a monthly newsletter, but also wish to be able to distribute an update via the **push technique** when very timely news occurs, such as a special event. In this case the provider can use pull updating in the communications object 110, but also include a receipt method that returns a registration message from the consumer the first time the communications object 110 is received. (Consumer registration information can be updated whenever the consumer changes it. Registration updates will be further discussed below.) These registration messages create a special association between the recipient 120 and communications object 110 which has a PushSpecial attribute (not shown in FIG. 3). Recipients 120 whose association with a communications object 110 has a PushSpecial attribute are ignored during standard communications object updates. However when the provider needs to distribute a push update, the provider can set a PushSpecial flag for the communications object 110 using the edit object form (322, FIG. 9). When this flag is set, all recipients 120 associated with the communications object 110 will receive the update via the **push technique**. Alternatively, the provider may choose to distribute a message object 110 to all recipients 120 who have a PushSpecial association with a communications object 110. This message object can include a receipt method that triggers an update via the pull technique. In this fashion a small message object may be distributed via a push medium such as **e-mail** in order to trigger the downloading of a much larger update via another medium such as the World Wide Web or FTP.

DETD Link control and update control have special applications to polymorphic service objects. The application of link control to polymorphic service objects is illustrated in FIG. 28. A provider using the provider program 12 has need of the services offered by a service object partner server 1302. First the provider obtains a service object 1310 from the partner server 1302 (step 1320). This may be by browsing with a web browser 50, receiving the service object 1310 via **e-mail**, or any of the other techniques described above. Such partner server 1302 may be a distribution server 32 or any other type of service object partner server such as those described below. Once the provider has obtained the service object 1310, the provider may add a link component object 110 to any of the provider's communications objects 110 which need to access the elements or methods the service object 1310. This link component object 110 will then be included in any communications object instance 35 generated from the consumer database 21 (step 1321). In a preferred embodiment, the link component object 110 is supplied by the service object 1310 itself. In this case, referring to FIG. 3, the provider need only create a contained-by association between the link component object 110 in the service object (1310, FIG. 28) and the provider's communications object 110. This association can also be created automatically when any service object method 141 is executed that creates a service relationship between the service object and the communications object 110. The communications object 110 thus becomes a synthesized object (813, FIG. 17), wherein the link component object 110 is supplied by the service object 1310. Examples of such a service object relationship include listing a communications object 110 in a directory server, registering a communications object 110 with an authentication server, or authorizing a communications object 110 for use with a payment server. Further examples will be given below. Referring again to FIG. 28, the next step is that a communications object instance 35 is transferred to a consumer program 22 (step 1322). This may be via **e-mail** using the **push**

technique, via a distribution server 32 using the pull technique, or any of the other techniques described above. Once the communications object instance 35 is transferred to the consumer program 22, a link method 141 of the link component object 110 may be manually executed by the consumer or automatically executed by another system method or communications object method. For example, the consumer may wish to look up related communications objects instances 35 in a directory server, or authenticate the communications object instance 35 before forwarding it, or make a payment transaction using the communications object instance 35. When the link method 141 is executed, it uses the attributes of the link element 143 to locate the designated service object 1310 as described in the communications object exchange control section above. For example, if the service object 1310 is not present locally in the consumer database 21, the link method uses other attributes of the link element to locate the service object 1310. For instance, if a URL was present, the link method would use it to obtain the service object 1310. If this fails, the link method would use the UID or name of the service object 1310 to obtain its URL or other current network address via a name server. The link method could also call the methods of a name service object, described below. Once the link method located the proper network address, it would download the service object 1310 from the partner server 1302 (step 1323). At this point the link is reestablished, and the communications object instance 35 can call the service methods of the service object 1310 to perform the services requested (step 1324).

DETD A second advantage to providers is that the distribution partner server 1302 can offload the work of transmitting communications object updates via the **push technique**. For large numbers of recipients 120, **e-mail** generation and transmission can require large amounts of computer processor time and network bandwidth. Offloading this to a distribution partner server 1302 can free the operation of the provider program 12 on a smaller personal computer. By maintaining a user object index at a distribution partner server 1302, the distribution partner server 1302 can also receive and process all acknowledgment message objects used to maintain the user object index for any communications object 110.

DETD Schedule coordination among any group is a fundamental challenge in communications. Scheduled events and schedule changes must be communicated to everyone in the group or else the group will not function. A communications object system can solve many widespread scheduling problems using a special type of communications object called a schedule object. Schedule objects are shown as class 817 of FIG. 17. Schedule objects 110 represent real-world events associated with any other communications object 110. Like all other communications objects 110, schedule objects 110 can contain schedule elements 143, schedule methods 141, and schedule rules 140 used to control scheduling operations. Collectively these are referred to as schedule control components. These all function as special cases of data exchange control, discussed above. Schedule objects 110 can also be nested as composite and component objects (811, 182, FIG. 17). This permits a composite schedule object 110 to contain multiple component schedule objects 110. An example would be a composite schedule object representing a multi-day **conference**, with component schedule objects representing the individual seminars at the **conference** . Like any communications object 110, schedule objects 110 can be distributed via push and pull and contain their own update methods. This is particularly relevant to schedule coordination because any changes to a schedule object 110 can be transmitted to all consumers associated with that schedule object 110 object. Using notification control, each user can also control exactly how he/she is notified of these changes. Schedule objects 110 can be maintained in any database 100, whether single-user or multiuser. Schedule objects 110 are particularly useful for managing group schedules in a multiuser database 100. This is because schedule objects 110 can easily be associated with user objects 110 or user group objects 110 to create and maintain scheduling relationship associations 111. Alternatively, communications object system programs can handle scheduling requests through an API with an

136/29-55

external scheduling program or database. A communications object system API is discussed further below.

DETD This technique for using schedule objects 110 can be generalized to many forms of scheduling. This includes business meetings; professional appointments; **teleconferences** or **videoconferences**; television, cable, or video shows; public seminars; and so on. The specific type of scheduling use is not a limiting feature of the invention.

141/40-46

	L #	Hits	Search Text
1	L1	446	(search same tool?).bsum.
2	L2	29	(search same tool? and www and internet).bsum.
3	L3	307293	709/200, 203, 217, 218, 225.ccls.
4	L4	1525	707/104, 004.ccls.
5	L6	458	707/501, 200.ccls.
6	L7	457	(707/501)
7	L8	948	707/200
8	L9	1236	345/326
9	L12	275	395/5$2.ccls.
10	L13	222	345/333.ccls.
11	L14	492	345/348.ccls.
12	L15	335	345/352.ccls.
13	L16	310969	3 or 4 or 5 or 6 or 7 or 8 or 9 or 10 or 11 or 12 or 13 or 14 or 15
14	L17	29	(1 and 2)
15	L19	0	(709/$.CCLS OR 707/$.CCLS OR 395/$.CCLS OR 345/$.CCLS OR 701/$.CCLS)
16	L20	56	((search same (tool? OR NAVIGATION OR ENGINE)) AND (FORMULAT? OR CONSTRUCT? OR EDIT?)).bsum.
17	L21	47	((EDIT? OR FORMULAT? OR CONSTRUCT?) AND (SEARCH SAME (SOFTWARE))).BSUM.
18	L22	8	20 AND 21
19	L23	0	(19 AND 22)
20	L18	20	17 and 16

	L #	Hits	Search Text
21	L24	15	(SEARCH AND (TOOL OR ENGINE OR SOFTWARE) AND (QUERY ADJ3 FORMULAT$)).BSUM.
22	L25	0	18 AND 24